MODELING PAST, CURRENT, AND FUTURE TIME IN MEDICAL DATABASES*

Vram Kouramajian[†]
Office of the Vice President
for Research & Information Systems
Rice University
P. O. Box 1892, Houston, Texas 77251
vram@rice.edu

Jerry Fowler
Medical Informatics &
Computing Research Program
Baylor College of Medicine
Houston, Texas 77030
gfowler@bcm.tmc.edu

ABSTRACT

Recent research has focused on increasing the power of medical information systems by incorporating time into the database system. A problem with much of this research is that it fails to differentiate between historical time and future time. The concept of bitemporal lifespan presented in this paper overcomes this deficiency. Bitemporal lifespan supports the concepts of valid time and transaction time and allows the integration of past, current, and future information in a unified model.

The concept of bitemporal lifespan is presented within the framework of the Extended Entity-Relationship model. This model permits the characterization of temporal properties of entities, relationships, and attributes. Bitemporal constraints are defined that must hold between entities forming "isa" hierarchies and between entities and relationships. Finally, bitemporal extensions are presented for database query languages in order to provide natural high-level operators for bitemporal query expressions.

INTRODUCTION

Humans naturally describe data in terms of time. Medical data are replete with temporal attributes, including diagnosis, prognosis, orders, duration of treatment, and appointment schedules. Modeling *temporal reality* with relational or object-oriented models leads to awkward modeling and unfriendly query languages. What is needed is a database system that uniformly integrates past, current, and future information in a single model.

Temporal databases preserve the complete history of the "Universe of Discourse;" that is, they follow the *non deletion* rule of data. This permits users to query the current state of the database, as well as past states, and even states that are planned for the future. Clinical patient records systems are a natural application of temporal databases due to the need for complete recall of patient history for clinical, legal, and research reasons [4, 9, 10]. Recent years have witnessed an increase in research on temporal databases [8, 12, 14]. (Excellent glossaries on temporal database concepts can be found in [1, 13]).

A problem with most research in temporal databases is inability to differentiate between historical time and future time. This is especially important for medical information systems where physicians frequently deal with future events such as testing and treatments, and modifications are frequently made to patient treatment schedules based on new information [9, 10]. Historical time is the time that an event *happened* in the real world. Future time is the time an event is scheduled or predicted to happen in the future. Campbell *et al.* [2] observe that expressing time as an interval is necessary to capture uncertainty in medical data. However, intervals alone cannot distinguish the past from the future. The differing semantics of historical time and future time should be reflected in the data model. The goal of this work is to introduce a temporal data model that uniformly integrates past, current, and future information.

A major contribution of this work is the development of the concept of *Bitemporal Lifespan,* which unites two main temporal concepts: valid time and transaction time. This new abstraction allows the integration of past, current, and future information in a unified model. We correlate bitemporal events with terms in English grammar to make the relationships between valid time and transaction time clear. We extend the Extended Entity-Relationship (*EER*) model by using bitemporal lifespans, and we introduce a natural temporal query language over bitemporal regions. We believe that it is more natural to specify temporal data and queries in a conceptual, entity-oriented data model than in a tuple-oriented, relational data model.

This work is an extension to our previous research in the area of temporal conceptual models and query languages [5], where a framework that differentiates between temporal and non-temporal objects was introduced. Here we propose a temporal *EER* conceptual model that distinguishes between the concepts of historical time and future time. Our model allows us to characterize the temporal properties of entities, relationships, and attributes. We also define temporal constraints that must hold between entities forming *isa* hierarchies and between entities and relation-

*Partially supported by National Library of Medicine Grant Number 1T15LM07093-02 and Texas Higher Education Coordinating Board-Advanced Technology Program.

[†]Current Affiliation: Department of Computer Science, The Wichita State University, Wichita, Kansas 67260, Email: vram@cs.twsu.edu.

ships. Finally, we propose bitemporal extensions for database query languages that provide natural high-level operators for bitemporal query expressions. For the sake of familiarity, we present these extensions in terms of *SQL*.

TIME ABSTRACTION

Representation of Clock Time

To represent *clock time*, let T be a countably infinite set of totally ordered discrete points in time, or *chronons*. A *time interval*, denoted by $[t_s, t_e]$, is defined to be a set of consecutive chronons; that is, the totally ordered set $\{t_s, t_{s+1}, \ldots, t_{e-1}, t_e\} \subset T$. We call t_s the *start time* and t_e the *end time* of the time interval.

Temporal granularity, the distance between two consecutive chronons, is application–dependent, and can be chosen as month, day, hour, minute, second, or any other suitable time unit. A single discrete chronon t is represented as an interval $[t,t]$, or simply $[t]$.

For historical databases, the domain of a valid time attribute is a time interval $[t_0, now]$, where t_0 represents the starting time of the database mini–world application, and *now* is the current time, which is continuously expanding. In a (general) temporal database, the valid time interval expands to cover the range $(t_{-\infty}, t_{+\infty})$, where values greater than *now* represent future data. In this case, the reference point t_0 is still employed, and negative subscripts are used to represent chronons that precede t_0.

Interval representation has an important shortcoming. Since the set of all intervals in T is not closed under set operations, Gadia and Yeung [7] suggested the concept of temporal elements. A *temporal element*, denoted as TE, is a finite union of time intervals, denoted by $\{I_1, I_2, \ldots, I_n\}$, where I_i is an interval in T. Union, intersection, and difference operations on temporal elements are easily defined. In addition, set comparison predicates of two temporal elements using $=$, $\neq$, $\supseteq$, $\supset$, $\subseteq$, and $\subset$ are also easily defined.

Bitemporal Lifespan

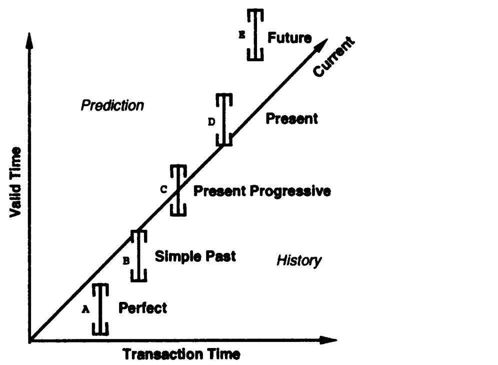
The *lifespan* of a database object is the time interval over which the object is defined. There are two types of lifespan, each of which is expressed in terms of clock time.

1. **Valid Time Lifespan:** The valid time lifespan of a database object is a temporal attribute that defines the clock time interval during which the object is deemed to be valid within the Universe of Discourse.

2. **Transaction Time Lifespan:** The transaction time lifespan of a database object refers to the clock time of timestamps associated with updates to the object by the database application.

Valid time lifespan refers to historical or predicted events, and can be determined by users of the system. Transaction time lifespan is associated with the recording of those events in clock time, and is determined solely by the system itself.

A *bitemporal lifespan* subsumes both valid time lifespan and transaction time lifespan. We represent a bitemporal lifespan BL as a finite set of tuples $BL = \langle VT, TT \rangle$, where VT is a temporal element in $(-\infty, +\infty)$ and TT is a time point in $[0, now]$. A *bitemporal chronon*, denoted by *bc*, is a point in the two-dimensional space $VT \times TT$; that is, $bc = \langle t_1, t_2 \rangle, t_1 \in VT, t_2 \in TT$. Our representation of bitemporal lifespans allows us to model past, current, and future information in a unified manner.



$VT = [t_1, t_2]; TT = t_3$				
Future	Present	Present Progressive	Simple Past	Perfect
$t_3 < t_1$	$t_3 = t_1$	$t_3 > t_1 \vee$ $t_3 < t_2$	$t_3 = t_2$	$t_3 > t_2$

Figure 1: Various Ways of Combining Valid Time with Transaction Time

Figure 1 shows the relationships between valid time and transaction time. The line marked "current" is the line where valid time and transaction time share the same clock time; that is, an event *e* was recorded at the same time the event occurred. To the right of this line is historical recording, in which an event is entered into the database *after* it occurred. To the left of this line lie predictions of the future, in which an event is entered into the database *before* it occurs. It is natural to describe these relations in terms of grammatical verb tense. Object A in Figure 1 is a historical fact recorded after its occurrence, and corresponds to the perfect tense: "The patient has had a previous pregnancy." Object B represents simple past tense. An event has just occurred, but is no longer active: "A drug was administered." Objects C and D both reflect present tense, because they indicate events recorded during the interval in which they are valid. Object C corresponds to present progressive tense, because it represents an ongoing activity, such as the course of a disease: "The patient is being treated for genital herpes." Object D corresponds to simple present tense: "We begin treatment now." Object E reflects the future. An event is recorded in the database before its occurrence: "The patient will return for an appointment next Tuesday."

THE BITEMPORAL DATA MODEL

The *EER* model has been extensively used in database design applications [3, 6]. In this section, we introduce extensions to the *EER* model that capture temporal data. For brevity, we assume familiarity with the basic concepts of the *EER* model and *EER* diagrams [3, 6], so that we may simply specify the novel concepts of the Bitemporal *EER* model.

Bitemporal Entities

An entity type is a set of entities of the same type; that is, entities that share the same attributes. Entities represent objects in the mini-world situation that is being modeled. Each entity type E_i has a set of attributes A_{i1}, A_{i2}, ..., A_{in}, and each attribute A_{ij} is associated with a domain of values $dom(A_{ij})$.

In the bitemporal *EER* model, each entity e of entity type E is associated with a bitemporal lifespan $BL(e)$ that defines the time during which e is valid.

Example 1: Figure 2 shows a fragmentary *EER* schema for a health care database, which includes the entity types *PATIENT*, *PHYSICIAN*, and *TEST*. A particular *PATIENT* entity may have a bitemporal lifespan $\{\langle[1/1/92, 1/2/93], 1/25/93\rangle, \langle[11/3/93, 2/5/94], 2/5/94\rangle\}$; this means that this patient was treated during the periods $[1/1/92, 1/2/93]$ and $[11/3/93, 2/5/94]$, and that this information was registered in the database at 1/25/93 and 2/5/94.

Bitemporal Assignment

The ***bitemporal assignment*** of each attribute A_i of an entity e, denoted as $A_i(e)$, is a partial function $A_i(e) : BL(e) \rightarrow dom(A)$. (This is similar to the idea of ***temporal assignment*** where the domain of a partial function is a temporal element [GaYe88].) We use $BL(A_i(e))$ to denote the subset of $BL(e)$ in which $A_i(e)$ is defined. The value of A_i during the time $BL(e) - BL(A_i(e))$ is undefined.

Example 2: Consider the database described by the schema in Figure 2. For simplicity, *day* is used as the temporal granularity. A particular *PATIENT* entity e_1 and a particular *PHYSICIAN* entity e_2 may have the following bitemporal attribute values:

$Name(e_1) = \{\ \langle[9/1/75, now], 1/15/94\rangle \rightarrow \textit{Jane Doe}\ \}$
$UID(e_1) = \{\ \langle[1/15/94, now], 1/15/94\rangle \rightarrow 123456789\ \}$
$Sex(e_1) = \{\ \langle[9/1/75, now], 1/15/94\rangle \rightarrow \textit{Female}\ \}$
$Problem(e_1) = \{\ \langle[1/15/94, 1/16/94], 1/15/94\rangle \rightarrow$
$\{\ \textit{Suspicion of Pregnancy}\ \}$,
$\langle[12/1/93, 8/1/94], 1/16/94\rangle \rightarrow$
$\{\ \textit{Pregnancy}\ \}\ \}$
$Name(e_2) = \{\ \langle[9/1/60, now], 1/15/90\rangle \rightarrow \textit{Yaa Dufie}\ \}$
$UID(e_2) = \{\ \langle[1/15/90, now], 1/15/90\rangle \rightarrow 987654321\ \}$
$Office(e_2) = \{\ \langle[1/15/90, now], 1/15/90\rangle \rightarrow$
$\{\ \textit{BCM \#315, BCM \#1015}\ \}\ \}$
$Specialty(e_2) = \{\ \langle[1/15/90, now], 1/15/90\rangle \rightarrow$
$\textit{Gynecology}\ \}$

Bitemporal Attributes and Keys

In our model, each entity has a system-defined, non-temporal, unique, immutable *SURROGATE* attribute (or "UID") whose value is not visible to users. In addition, several types of bitemporal attributes exist:

1. **Bitemporal Single-Valued Attribute:** A bitemporal single-valued attribute has at most a single atomic value for each entity at each bitemporal instant *bc*.

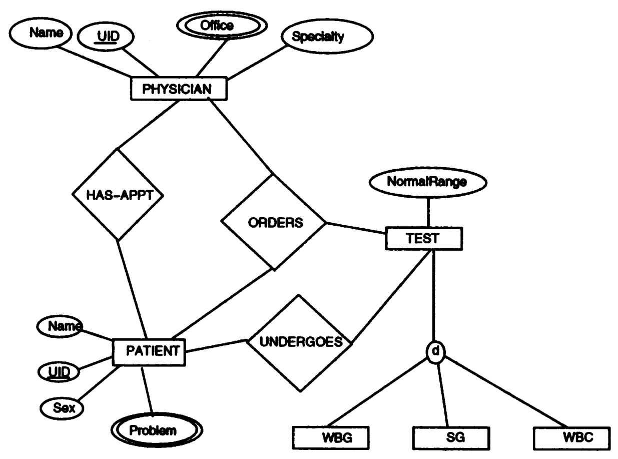


Figure 2: A Schema Fragment

2. **Bitemporal Multi-Valued Attribute:** A bitemporal multi-valued attribute can have more than one value for an entity at a given bitemporal instant *bc*; hence, its domain is the power set $P(V)$ of some simple domain V. For instance, in Figure 2, *Problem* is a multi-valued attribute since a patient may have more than one problem concurrently. *Office* is likewise multi-valued.

3. **Bitemporal Composite Attribute:** A bitemporal composite attribute is a list of several component bitemporal attributes, and its value for each entity at bitemporal instant *bc* is a concatenation of the values of its components.

4. **Key Attribute:** A (simple or composite) attribute A is a key attribute of an entity type E if at any bitemporal instant *bc*, no two entities in E have the same value for A. We allow the update of a key attribute since each entity is uniquely identified by its *SURROGATE*. For instance, the attribute *UID* of entity Type *PATIENT* in Figure 2 is a key attribute since it uniquely identifies a particular patient.

Bitemporal Relationships

A ***relationship type*** R of degree n has n participating entity types $E_1, E_2, ..., E_n$. Each ***relationship instance*** r in R is an n-tuple $r = \langle e_1, e_2, \ldots, e_n\rangle$ where each $e_i \in E_i$. In our model, each relationship instance r is associated with a bitemporal lifespan $BL(r)$. The constraint is that $BL(r)$ must be a subset of the intersection of the bitemporal lifespans of the entities $e_1, e_2, \ldots, e_n$ that participate in r. That is, $BL(r) \subseteq (BL(e_1) \cap BL(e_2) \cap \ldots \cap BL(e_n))$. This is because for the relationship instance to exist at some bitemporal instant *bc*, all the entities participating in that relationship instance must also exist at *bc*.

Relationship attributes are treated similarly to entity attributes; the bitemporal value $A_i(r)$ of each simple attribute A_i of r is a partial function (bitemporal assignment) $A_i(r) : BL(r) \rightarrow dom(A_i)$ and $BL(A_i(r)) \subseteq BL(r)$.

Example 3: In Figure 2, the relationship type *HAS_APPT* represents the fact that a patient has an appointment with a physician; the relationship type *ORDERS* represents the fact that a physician orders a test for a patient; and the relationship type *UNDERGOES* represents the fact that a patient undergoes a test. To Example 2, let us add a *HAS_APPT* instance r between the *PATIENT* entity e_1 and the *PHYSICIAN* entity e_2, with $BL(r) = \{\langle[1/15/94], 1/15/94\rangle, \langle[1/16/94], 1/15/94\rangle\}$; this indicates that the patient had a walk-in appointment on 1/15/94 and at that time a second visit was scheduled for 1/16/94 to discuss test results.

Bitemporal Hierarchies

Subclasses can be used to represent generalization and specialization hierarchies and lattices [6]. Membership of entities in a subclass can either be specified via a predicate or categorized explicitly by the user. In the former case, we have a ***predicate-defined subclass***, where each entity in the superclass that satisfies a defining predicate will be a member of the subclass. In the latter case, the user ***explicitly*** partitions entities from the superclass into categorical subclasses.

An entity e of a superclass C will belong to a predicate–defined subclass SC at all bitemporal instants where the predicate evaluates to true. For a user defined subclass, the user will specify the bitemporal instants at which an entity e belonging to the superclass C will also belong to the subclass SC. In either case, the entity will have a bitemporal lifespan $BL(e/SC)$ that specifies the bitemporal instants at which it is a member of the subclass SC. The constraint $BL(e/SC) \subseteq BL(e/C)$ must always hold.

Example 4: Figure 2 shows an example of three subclasses: *WBG* (*Whole Blood Glucose*), *SG* (*Serum Glucose*), and *WBC* (*White Blood Count*) of the *TEST* entity type. The symbol d in Figure 2 indicates that the subclasses *WBG*, *SG*, and *WBC* are always bitemporally disjoint.

THE BITEMPORAL ALGEBRA

To allow for bitemporal constructs in queries, we define the concepts of bitemporal boolean expressions, bitemporal selection conditions, and bitemporal projections.

Bitemporal Boolean Expression

A bitemporal boolean expression is a conditional expression on the attributes and relationships of an entity. A bitemporal boolean condition c_t, when applied to an entity e at a given transaction time point t, evaluates to { *TRUE, FALSE, UNKNOWN* }.

Example 5: The boolean expression (Problem = "Pregnancy")$_{1/16/94}$ for the *PATIENT* entity e_1 given in Example 2 results in:

$\{ [12/1/93, 8/1/94] \rightarrow TRUE,$
$[9/1/75, 11/30/93] \cup [8/2/94, now] \rightarrow FALSE,$
$other\text{-}times \rightarrow UNKNOWN\}$

However, (Problem = "Pregnancy")$_{1/1/94}$ results in:

$\{ [9/1/75, now] \rightarrow FALSE,$
$other\text{-}times \rightarrow UNKNOWN\}$

True_Time

The true_time of a boolean expression $[\![c_t]\!]$ at a given transaction time point t, evaluates to a ***temporal element*** for each entity e. The temporal element is the time for which the condition is *TRUE* for e.

Example 6: The true_time of the boolean condition in example 5 evaluated at the transaction time point 1/16/94 is { [12/1/93, 8/1/94] }.

Bitemporal Selection Condition

A bitemporal selection condition compares two true_time expressions (i.e. temporal elements) using the set comparison operators $=$, $\neq$, $\supseteq$, $\supset$, $\subseteq$, and $\subset$. When applied to an entity type, it evaluates to those entities that satisfy the bitemporal selection condition.

Example 7: Consider the following bitemporal selection condition applied to the *PATIENT* entity type of Figure 2:

$$[\![\text{(HAS\_APPT = "Yaa Dufie")}_{1/15/94}]\!] \supseteq [1/1/94, now]$$

This selects the ***entire history***, as registered in the database on 1/15/94, of all patients who had appointments with Yaa Dufie between 1/1/94 and the present.

Bitemporal Projection

A bitemporal projection TE_t of a bitemporal entity e over a temporal element TE at transaction time t, is evaluated in two steps: (1) rollback to the transaction time t and (2) restrict the data displayed for the entity e to the temporal element TE. (This operation is similar to the when operator introduced in [GaYe88] for clock time periods.)

Example 8: The bitemporal projection of bitemporal *PATIENT* entity e_1 of Example 2 over the temporal element $\{[2/15/94, now]\}$ at the transaction time point 2/15/94 results in:

$Name(e_1) = \{ [2/15/94, now] \rightarrow Jane\ Doe \}$
$UID(e_1) = \{ [2/15/94, now] \rightarrow 123456789 \}$
$Sex(e_1) = \{ [2/15/94, now] \rightarrow Female \}$
$Problem(e_1) = \{ [2/15/94, now] \rightarrow \{Pregnancy\} \}$

THE BITEMPORAL QUERY LANGUAGE

In non–temporal databases, a query will typically select certain entities based on boolean predicates that involve attribute values of the entity and of related entities, and then display certain attributes or relationships of each of the selected entities. In a bitemporal database, selection criteria may be based not only on attribute values but also on bitemporal conditions. In addition, once an entity is selected, a user may wish to display the complete history (bitemporal assignment) of some of its attributes or relationships, or to limit the displayed values to a certain time interval.

Our bitemporal algebra can be used to specify temporal queries by extending the *SQL* database language [6]. The bitemporal extensions are illustrated through examples (A number of important bitemporal query constructs, such as bitemporal aggregates, are omitted for brevity).

Example 9: Consider the query to retrieve as of 1/1/92 (transaction time) the name and office number of all pediatricians in the clinic on 1/1/88 (valid time):

SELECT ⟨ Name, Office⟩ : $[1/1/88]_{1/1/92}$
FROM PHYSICIAN
WHERE $[\![$(PHYSICIAN.Specialty = "Pediatric")$_{1/1/92}]\!]$
$\supseteq$ [1/1/88]

It is important to distinguish between the effects of the bitemporal construct used in the SELECT–clause and that in the WHERE–clause. The WHERE–clause evaluates entities based on the state of the database on 1/1/92. It is still necessary to specify the bitemporal projection $[1/1/88]_{1/1/92}$ again in the SELECT–clause in order to obtain the information registered on 1/1/92 rather than the current information.

Example 10: Consider the query to retrieve the current name and office number of all physicians who saw the patient "Jon Smith" during the time period [1/1/88, 4/5/92] as registered in the database on 5/5/92:

SELECT ⟨Name, Office⟩
FROM PHYSICIAN
WHERE $[\![$(PHYSICIAN.HAS_APPT.PATIENT.Name =
"Jon Smith")$_{5/5/92}]\!]$ $\supseteq$ [1/1/88, 4/5/92]

To deal with bitemporal data, we need bitemporal query functions not found in traditional databases [11]. The functions *FI* (first instant) and *LI* (last instant) return the first and last time points, respectively, when applied to a temporal element. Other functions we will use are the *VTIME* and *TTIME* functions in the *SELECT* clause, which retrieve the valid time and transaction time of each selected entity respectively.

Example 11: To retrieve the interval during which "Yaa Dufie" was on the staff as of 1/1/94, we write:

SELECT ⟨VTIME⟩
FROM PHYSICIAN
WHERE PHYSICIAN.Name = "Yaa Dufie"

CONCLUSIONS

In this paper, we presented a bitemporal extension to the *EER* model that uniformly supports past, current, and future times. The concept of bitemporal lifespan of an entity or a relationship instance was defined. The bitemporal properties of entities, relationships, and attributes were characterized. Bitemporal constraints that must hold between entities forming *isa* hierarchies as well as between entities and relationships were defined. We also presented the concepts of bitemporal selection conditions and bitemporal projections, and showed how these bitemporal constructs can be used to extend the *SQL* database language.

We believe that the use of the bitemporal *ER* model will provide greater power in the expression of medical queries. We hope to prototype a bitemporal *DBMS* that addresses important implementation issues like bitemporal indexing structures and query optimization. However, temporal databases will not become ubiquitous unless they are able to provide *human-centric* user interfaces over terabytes of temporal data. Needless to say, this is an immensely challenging area worth exploring.

References

[1] K. Al-Taha, R. Snodgrass, and M. Soo. Bibliography on spatiotemporal databases. *ACM SIGMOD Record*, 22(1), March 1993.

[2] K. Campbell, A. Das, and M. Musen. A logical foundation for the representation of clinical data. *Journal of the American Medical Informatics Association*, 1(3), May/June 1994.

[3] P. Chen. The entity–relationship model — towards a unified view of data. *ACM Transactions on Database Systems*, 1(1), March 1976.

[4] A. Das, S. Tu, G. Purcell, and M. Musen. An extended sql for temporal data management in clinical decision-support systems. In *Symposium on Computer Applications in Medical Care*, November 1992.

[5] R. Elmasri and V. Kouramajian. A temporal query language for a conceptual model. In N. Adam and B. Bhargava, editors, *Advanced Database Systems, Lecture Notes in Computer Science*, volume 759. Springer–Verlag, December 1993.

[6] R. Elmasri and S. Navathe. *Fundamentals of Database Systems, 2nd Edition.* Benjamin/Cummings, 1994.

[7] S. Gadia and C. Yeung. A generalized model for a temporal relational database. In *ACM SIGMOD International Conference on Management of Data*, June 1988.

[8] C. Jensen et al. A consensus glossary of temporal database concepts. *ACM SIGMOD Record*, 23(1), March 1994.

[9] M. Kahn, L. Fagan, and S. Tu. Extensions to the time–oriented database model to support temporal reasoning in medical expert systems. In *Methods of Information in Medicine*, 1991.

[10] Y. Shahar, S. Tu, and M. Musen. Temporal–abstraction mechanisms in management of clinical protocols. In *Symposium on Computer Applications in Medical Care*, November 1992.

[11] R. Snodgrass. The temporal query language *TQUEL*. *ACM Transactions on Database Systems*, 12(2), June 1987.

[12] R. Snodgrass, editor. *International Workshop on an Infrastructure for Temporal Databases*, June 1993.

[13] M. Soo. Bibliography on temporal databases. *ACM SIGMOD Record*, 20(1), March 1991.

[14] A. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. Snodgrass, editors. *Temporal Databases: Theory, Design and Implementation.* Benjamin/Cummings, 1993.